Proceedings of the Design Society, Volume 6: DESIGN 2026
https://doi.org/10.1017/pds.2026.10598

Automatic feature recognition from imperfect models using a novel workflow of data surrogation

Aman Kukreja[1,✉], Chris Cox[1], James Gopsill[1], Kristin Paetzold-Byhain[2] and Chris Snider[1]

[1] University of Bristol, United Kingdom, [2] Dresden University of Technology, Germany

✉ aman.kukreja@bristol.ac.uk

ABSTRACT: Imperfect CAD models with non-smooth features are common outputs of the latest digital tools. These are unsuitable for the feature recognition needed for end applications like computer-aided manufacturing. This paper proposes to recognise features from imperfect models by contributing a comprehensive dataset, a novel data surrogation method, and ML-based automated feature recognition model. Results show that the data surrogation method accurately replicates manual imperfections with voxel accuracy >0.9 and a Dice coefficient >0.6. Ultimately, feature recognition achieves 92.8% test accuracy.

KEYWORDS: artificial intelligence (AI), computer-aided design (CAD), computational design methods

1. Introduction

Conventional CAD modelling is widely used for designing engineering products from scratch. In recent years, due to increased focus on rapid prototyping and reverse engineering, alternative methods of rapid design are being explored. These include Virtual reality-based CAD modelling solutions (Kukreja et al., 2026; Prinz et al., 2025), reconstruction of existing parts or worn-out parts using laser scanning (Kukreja et al., 2025), generative AI-based design (Wu et al., 2023), and image-based reconstruction methods (e.g. photogrammetry and Gaussian splatting). The output of these methods is in the form of meshes (STL), point clouds, or voxel-based representations, which are often processed further to generate parametric models to make them suitable for downstream applications, such as finite element analysis (FEA), Computer-Aided Process Planning (CAPP), and product lifecycle management (PLM) search and query systems.

Creating parametric models using these output formats requires Feature Recognition (FR), which involves identifying and extracting application-specific regions of interest, such as machining features, from the input geometry (Pande, 2011). However, the output formats provided by these methods often have structural inconsistencies and noise (termed imperfections in this work). These inconsistencies can occur due to improper scanning and occlusions for laser scanners, inconsistencies in mesh generation algorithms (Fig. 1(b)), through applications such as VR-based CAD modelling, or by scanning worn-out/defective parts (Fig. 1(c)), resulting in inaccurate/non-manifold output models (Freeman Gebler et al., 2021). These output models with imperfections are referred to here as imperfect models, and they are unsuitable for existing rule-based or Machine Learning (ML) based methods. Traditionally, this failure necessitates a cumbersome, error-prone preprocessing pipeline in which models are manually or semi-automatically cleaned to remove invalidities (such as noise or holes) and undergo explicit feature reconstruction before feature recognition can even be attempted. This reliance on pre-processing creates a bottleneck in rapid design workflows.

To overcome the inefficiency of the traditional approach of digital design workflows, i.e. pre-processing and reconstruction before FR from imperfect models, this paper provides a novel dataset and its creation workflow to enable direct recognition of features from imperfect CAD models. The paper thus makes two main contributions: first, the creation of a comprehensive dataset of imperfect CAD models for machine learning-based feature recognition, and second, a general, novel ML-based data surrogation technique that can be adapted for other methods of model reconstruction. We used voxel-based models in this work,

https://doi.org/10.1017/pds.2026.10598 Published online by Cambridge University Press

as they leverage a volumetric data structure that is particularly well-suited for handling the complex, irregular three-dimensional features found in imperfect models.

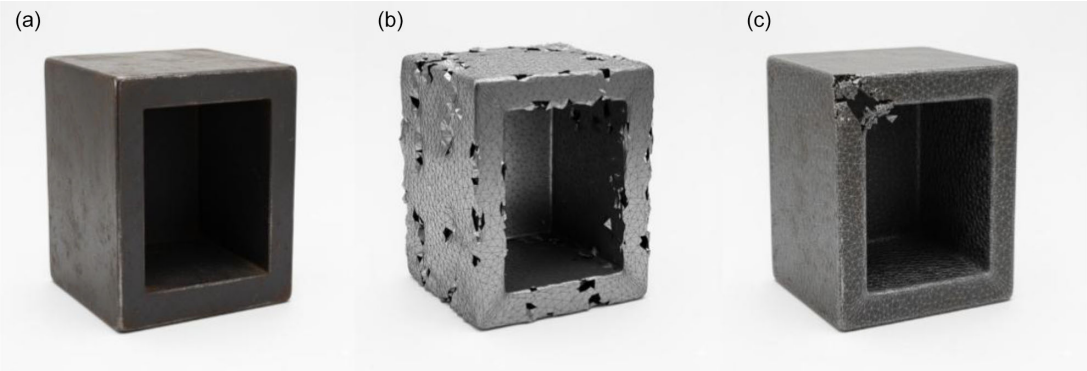


Figure 1. Imperfections in features (a) real, (2) meshing defects (c) scan of defective part

The remainder of this paper is organised as follows: Section 2 reviews the state-of-the-art literature regarding feature recognition and deep learning methodologies. Section 3 details the proposed data surrogation technique using the U-Net and network architecture of the Automatic Feature Recognition (AFR) model. Section 4 presents the results of the U-Net and tests the AFR model on imperfect and perfect datasets, and Section 5 discusses their implications and the future scope. Finally, Section 6 gives the important conclusions of the work.

2. Literature review

Automatic Feature Recognition (AFR) is a well-researched area with many different methods proposed over the last three decades. This includes heuristic-based, graph-matching-based, and learning approaches. Heuristics-based methods rely on predefined geometric rules for each feature and are time-consuming to set up, lacking generalizability across the diverse range of manufacturing features encountered in modern designs. In comparison, graph-based approaches are more successful with some degree of generalisability (Sunil & Pande, 2009). While effective for well-defined features, both these methods would struggle to recognise features from imperfect models without reconstruction. This is because such models may have disjoint edges, ill-defined faces and edges, and non-manifold geometry, all of which would be hard to bind into rules or form a consistent structure for the graph. Graph-based approaches are known to struggle with recognising intersecting features, further suggesting their inability to handle imperfect models (Z. Zhang et al., 2018).

On the contrary, ML-based methods are more generalisable, provided there is enough data for training. Various data formats have been used as inputs for these methods, including B-rep models, adjacency graphs, point clouds, and voxels, each with its own pros and cons. For instance, (Cao et al., 2020) used B-rep models to train a Graph Neural Network for automated recognition of solids with planar surfaces. (Rameshbabu & Shunmugam, 2009) used face adjacency graphs to recognise manufacturing features using STEP files. (Sunil & Pande, 2009) used B-rep models, pre-processed them to generate feature representation vectors that are then used to train an Artificial Neural Network (ANN) to recognise features. All these methods, however, require elaborate geometry and topological representations to work. On the contrary, methods based on point clouds and voxel-based data are more flexible and compatible with approximate geometric models (Vatandoust et al., 2025). (H. Zhang et al., 2022) used point cloud data for feature recognition and proposed a novel Associatively Segmenting and Identifying Network that could segment intersecting features and recognise features simultaneously. They used uniform point cloud data from BRep models to train a CNN for AFR. (Z. Zhang et al., 2018) created the FeatureNet dataset and proposed a 3D convolutional neural network to recognise 24 manufacturing features using voxel-based CAD model data. They created a comprehensive dataset of uniform, perfect models synthetically and demonstrated the effectiveness of voxel-based AFR in effectively recognising intersecting features. (Vatandoust et al., 2025) proposed a computationally efficient manufacturing feature recognition method using sparse voxel representation and creating an octree-based sparse voxel CNN. They achieved a high accuracy while reducing GPU memory consumption by 44%, showing the promising performance of voxel-based AFR methods. Although ML-based AFR methods claim to work with slightly noisy data, such as scanned data, their effectiveness has not yet been formally tested. Nonetheless, with excessive noise, as in

https://doi.org/10.1017/pds.2026.10598 Published online by Cambridge University Press

VR-based outputs involving human hand movements in real 3D space, or digital camera-based scanning methods with lower-resolution cameras, or laser scans of worn-out parts, these methods are likely to perform poorly, and a need thus exists to investigate this aspect and propose a solution.
To summarise, the existing AFR techniques demonstrate the ability to understand spatial contexts and recognise features from varied input forms with high accuracy. However, no method has been developed to recognise machining/design features from the imperfect CAD models. This paper proposes a novel method to curate the imperfect CAD model data and use it to train an exemplar 3D CNN-based AFR model.

3. Methodology

The approach taken in this work to address the identified problem of imperfect models in digital design workflows is rooted in a broader body of research on extracting features from VR-sculpted inputs and subsequently utilising them automatically to create parametric models. Despite its intuitiveness and alignment to real-world sculpting, human physical input in the digital world is inherently imprecise. This creates both a challenge and an opportunity - engineering CAD systems aiming to operate in Virtual Reality with human input must contend with high degrees of input imprecision, necessitating work on ML recognition as previously published and presented here (Kukreja et al., 2026). However, the imprecise models produced then serve as a randomised collection of errors in geometries while preserving their spatial construct. Consequently, addressing this problem could be considered fundamentally the same as feature extraction from scan data and reverse engineering, which also have geometric inconsistencies and imperfections. Using this approach of VR sculpting to create imperfect models could thus allow the creation of a labelled dataset and, consequently, ML training for the present work, as explained below.
The methodology comprises three steps: i) Perfect-Data curation, ii) Imperfect-Data generation and augmentation, and iii) ML-based AFR model training. These are explained as follows.

3.1. Perfect-data curation

Industrial products exhibit a wide range of design features, primarily comprising machining features and freeform surface features. In this work, we focus on prismatic components, with our primary application being the design of parametric models using VR-sculpting outputs. Such geometries are also common in industrial systems due to their innate manufacturability, and their 2.5D topology serves as a suitable case for proof-of-principle investigation. Therefore, we concentrate on machining features. However, a similar data generation technique can also be applied to other applications.
The base dataset of perfect CAD models is drawn from the FeatureNet dataset, a widely used benchmark for testing state-of-the-art ML models for feature recognition tasks, which consists of 24 commonly occurring manufacturing feature types (Vatandoust et al., 2025; Z. Zhang et al., 2018). It consists of 24 machining features, of which 22 features were taken and categorised into 7 main design features, as shown in Table 1. Two features, i.e., Chamfer and Round, were not considered at this point as they were deemed secondary design features. FeatureNet models are mesh-based models and were voxelised using ray tracing. The voxel-based models are represented in a 3D grid of resolution 64,64,64 containing 0s and 1s, where 0s represent the absence of material (suggesting the presence of a feature) and 1s represent the

Table 1. Design feature classification from manufacturing features

S. no.	FeatureNet Category	Assigned Feature Category
1	O ring	O ring
2	Circular end pocket	Circular end pocket
3	Vertical and Horizontal Circular end blind slot, Circular blind step	Circular end blind slot
4	Through hole, Blind hole	Circular pocket
5	Two sides through step, slanted through step, Six sides passage, Six sides pocket	Non-rectangular pocket
6	Rectangular passage, Rectangular through slot, rectangular blind slot, Rectangular pocket, Rectangular blind step, Rectangular through step, Two sides through step	Rectangular pocket
7	Triangular passage, triangular pocket, Triangular blind step, Triangular through slot	Triangular pocket

https://doi.org/10.1017/pds.2026.10598 Published online by Cambridge University Press

presence of material. It must be noted that FeatureNet models were generated using CAD modelling software and are defect-free (referred to as perfect in this work).

3.2. Imperfect data generation and augmentation

The second step is to generate the imperfect CAD model data. Figure 3 illustrates the data generation pipeline employed in this work to create a comprehensive dataset with diverse variations, suitable for training a deep learning network. It begins with the base imperfect models, manually generated by the authors, followed by augmenting the models through shrinkage and shifting. The generated models are then used to train a U-Net that learns the pattern of imperfections generated manually. Finally, the trained U-Net is used to generate the final dataset of imperfect models using perfect (FeatureNet) CAD models as inputs. All these steps are explained below.

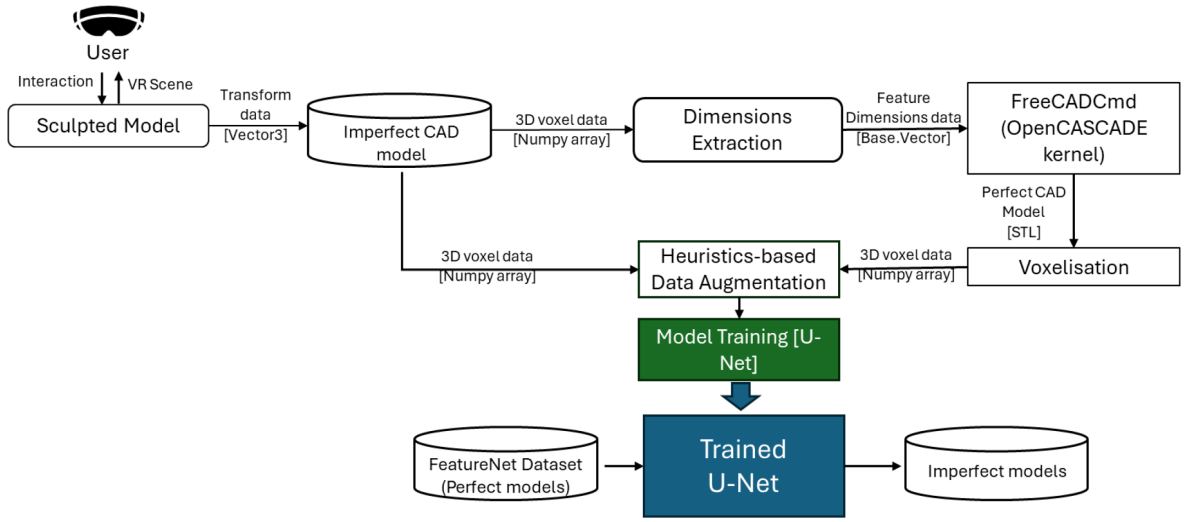


Figure 2. Data generation pipeline

3.2.1. Manually generated imperfect models

Base imperfect CAD models were generated by the authors using a VR-based digital sculpting tool developed in-house (Fig. 2). It lets users sculpt and remove material from the raw stock to create design features using VR controllers. After sculpting, a button allows the user to export the current state of the model as a CSV. This CSV contains the location of the removed voxels, which is then converted into a voxel grid. A total of 20 models were generated for each feature using this tool in 2 hours by the authors, creating a total of 140 models.

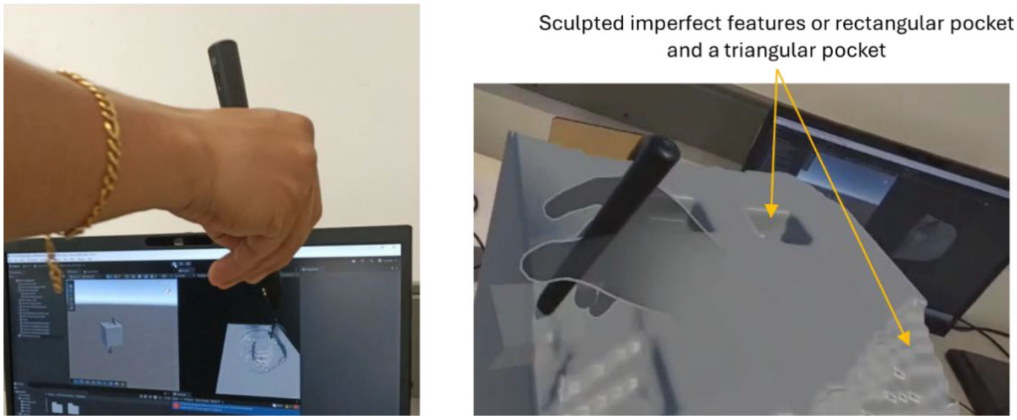


Figure 3. Manual imperfect data generation using VR-based digital sculpting

3.2.2. Data Augmentation

Deep learning requires a large amount of labelled data. However, generating this much data is a cumbersome task. Therefore, we devised a method to learn to create human-like imperfections using deep learning. Once learnt, this can be used to take perfect models as input and generate corresponding imperfect

https://doi.org/10.1017/pds.2026.10598 Published online by Cambridge University Press

models that preserve the location and overall dimensions of the feature, but with noisy boundaries, as a human would create in reality and as is common in scanned data inputs such as point clouds.
The task was thus to create a function $\boldsymbol{f}_x$ operating on a perfect dataset of voxel-based models ($\boldsymbol{D}_p$) to produce an imperfect dataset ($\boldsymbol{D}_i$). It was envisaged that a supervised machine learning (ML) technique would be used to do this, given its stochastic nature and ability to learn complex patterns.

$$f_x\left(D_p\right) = D_i$$

To train the ML model, a dataset of manually generated imperfect data, along with its corresponding perfect data, was generated using four steps (shown in Fig.3): i) Analyse the imperfect manually-generated voxel-based CAD models to deduce the dimensions and locations of the features using computational algorithms, ii) Generate perfect models using the extracted dimensions in a CAD modelling software and export them as mesh-based models, iii) Voxelise the mesh-based model to generate a perfect voxel-based CAD model, iv) Heuristics-based data augmentation.
The first two steps were implemented using the Python-based FreeCAD API (freecadcmd). The dimensions were extracted by fitting the desired feature shape onto the voxelised models using heuristics/OpenCV-based functions, which were then passed to FreeCAD to automatically generate the parametric models and export them as mesh (STL) models. Finally, the STL model was voxelised using the ray casting algorithm. The perfect-imperfect pairs of models generated after voxelisation were then augmented to generate surrogate data for training a deep learning network for learning imperfections. It includes two steps: i) Shrinkage, and ii) Shifting. Shrinkage is performed by first interpolating the features within each 2D layer using random voxel units, utilising the zoom function in scipy, followed by clipping and padding to retain the original size (i.e., 64, 64). Whereas, shifting moves the feature along x and y directions by random voxel units and creates 10 new models for each feature type. This entire pipeline outputs a set of perfect voxel-based CAD models corresponding to imperfect voxel-based CAD models that form imperfect-perfect pairs. In this work, a total of 700 perfect-imperfect pairs were generated for each category.
The perfect-imperfect pairs of models are used to train the ML network to learn the imperfections of the VR-based sculpting for each feature type. In this work, a 3D CNN with U-Net architecture was used. U-Net is widely used for image segmentation, learn contextual information, and lately in diffusion models to perform denoising. In our case, we need reverse functionality, i.e. add noise in the data such that it replicates randomness. U-Net was chosen because it can efficiently capture contextual information using an encoder-decoder structure, accurately reconstruct voxel-level details using up-sampling, maintain spatial precision of features using skip connections and same input-output resolutions, and learn local noise patterns using convolutional filters.
A total of 7 different U-Net were trained and validated corresponding to 7 design feature categories. The U-Net architecture used in this work is illustrated in Fig. 4. It comprises four levels for the Encoder-Decoder and a bottleneck at its centre. Each Encoder level consists of Conv3D layers along with down-sampling using Max-pooling, and the Decoder level consists of ConvTranspose for up-sampling and skip

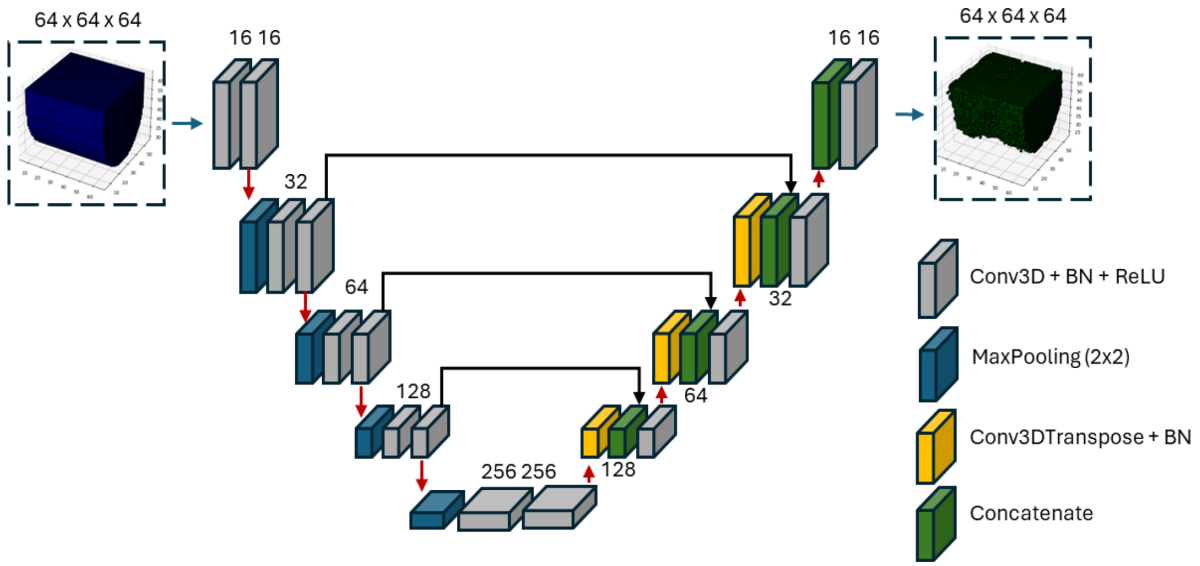


Figure 4. U-Net achitecture for learning imperfection patterns

https://doi.org/10.1017/pds.2026.10598 Published online by Cambridge University Press

connections to concatenate feature maps from corresponding encoder levels. The model takes a perfect voxel-based CAD model (size: 64, 64, 64) as input and outputs an imperfect model (size: 64, 64, 64). For supervised learning, pairs of perfect and imperfect models generated above were used. The training objective of the U-Net was to minimise reconstruction loss ($\boldsymbol{L}$) given by:

$$L = \left\| f_x(D_p) - (D_i) \right\|^2$$

The trained U-Net was tested to determine whether it preserved spatial context and created a reasonable overlap with the original perfect data (Section 4.2). The final trained U-Net were then used to convert perfect FeatureNet models (1000 per category) into imperfect models (1000 per category). This, along with the initial augmented dataset (i.e., imperfect and perfect models used for training the U-Net), forms the final augmented dataset (2400 per category). This results in a total dataset of 3,400 models per category, including perfect models.

3.3. Automatic feature recognition using imperfect models

The comprehensive dataset of perfect and imperfect models created by the above data generation and augmentation pipeline was used to train an AFR model. A bespoke 3D Convolutional Neural Network (CNN) was created to do this task and test the validity of the dataset. Other state-of-the-art networks can also be used to create a feature recognition system for imperfect models using the presented dataset.
Three networks with the same architecture were trained on: D1 Combined perfect and imperfect models (3400 per category), D2 Perfect models only (3400 per category augmented by mirroring 1700 perfect models), and D3 Imperfect models only (3400 per category augmented by mirroring 1700 imperfect models). The network was optimised via hyperparameter tuning to obtain optimal parameters for the given datasets. Figure 5 shows the architecture of the bespoke 3D CNN. It has four convolutional layers, one dense layer, and a total of 728,775 parameters. The network trained on D1, D2, and D3 achieved validation accuracies of 99.42%, 99.33%, and 99.52%, respectively. These trained models were tested on a previously unseen/new manually generated dataset, the results of which are presented in the following section.

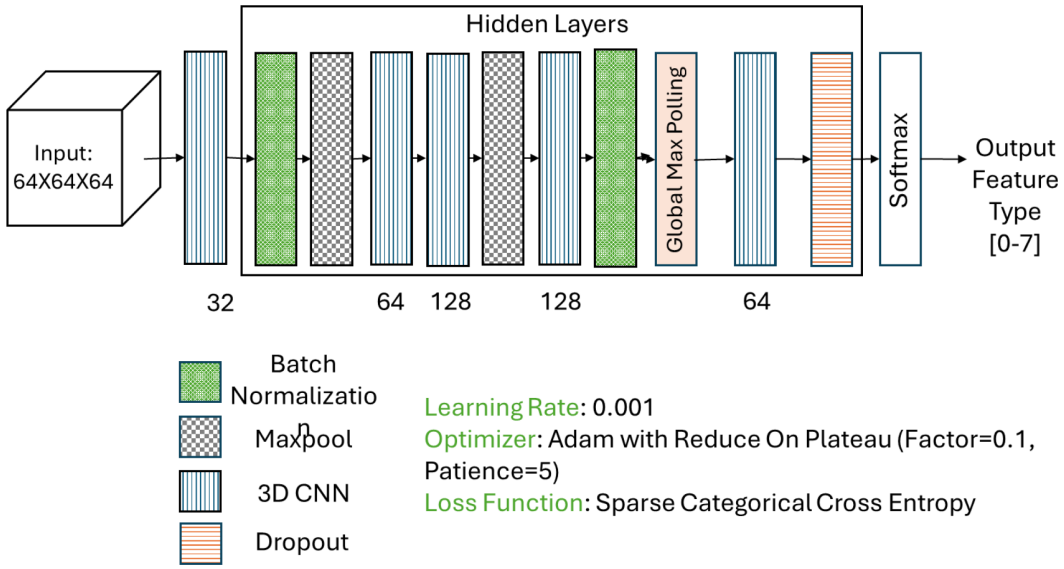


Figure 5. Bespoke 3D CNN architecture

4. Results

This section presents the results of i) Data surrogation and augmentation network (U-Net), and ii) Automated Feature Recognition model (3D CNN).

4.1. Results of U-Net-based surrogation

The trained U-Net was tested both qualitatively (visually) and quantitatively (metric-based). The output imperfect models generated by feeding perfect (FeatureNet) models, were used for evaluations. The U-Net gives the full 3D replication of perfect models with noise. For qualitative assessment, 100 of 1000 samples were plotted and assessed. Figure 6 shows sample perfect-imperfect pairs for all feature types. Only the top

https://doi.org/10.1017/pds.2026.10598 Published online by Cambridge University Press

layer of each feature type is shown here to demonstrate the successful reproduction of imperfections while maintaining the topological and spatial information of the feature. Figure 7 shows 3D voxel-based models produced by the U-Net for two exemplar categories: triangular pocket and circular end slot.

For quantitative evaluation, two standard parameters, the Dice coefficient and Voxel accuracy, were used. Dice coefficient (Eq. given below) is a standard measure of overlap between predicted output (i.e. imperfect voxel-based model) and the ground truth (i.e. perfect voxel-based model). Whereas voxel accuracy (Eq. given below) is the fraction of correctly predicted voxels (both 1s and 0s) over all voxels, which shows whether geometry is preserved while defects are added or not.

$$Dice = \frac{2TP}{2TP + FP + 2FN}$$

$$Voxel\ accuracy = \frac{TP + FN}{TP + FP + FN + TN}$$

Here, TP, FP, TN, and FN are true and false positives and negatives, respectively.

Table 2 shows the evaluated Dice coefficients and Voxel accuracies of perfect-imperfect model pairs for the manually generated dataset (D_o) and the U-Net generated dataset (D_u) for each of the 7 categories. Voxel accuracy of the trained U-Net for each feature was above 0.92 and within 0.25%-4% of the D_o, showing that the U-Net was able to preserve the shape and size of the features. Dice coefficients for D_u were above 0.6 for all features, showing a good similarity between the input and output and that a substantial amount of noise has been added. The values of the Dice coefficients were within 15% of those of the training dataset, indicating that the trained U-Net was able to replicate the manually generated noise in the perfect FeatureNet dataset. In essence, a lower Dice score with a higher voxel score implies that the model is similar, but with a high degree of noise.

Table 2. Quantitative evaluation of U-Net to generate imperfect voxel-based models

	Voxel Accuracy		Dice Coeff.	
Feature Type	**Manual**	**FeatureNet**	**Manual**	**FeatureNet**
O-Ring	0.973	0.9684	0.6087	0.639
C-End	0.98	0.9232	0.81	0.8603
V-Cend	0.982	0.9602	0.8385	0.8454
C-Pock	0.9689	0.9419	0.8089	0.805
NR	0.9063	0.9441	0.7021	0.6135
Rec	0.9833	0.9757	0.904	0.9307
Tri	0.9689	0.9578	0.6298	0.729

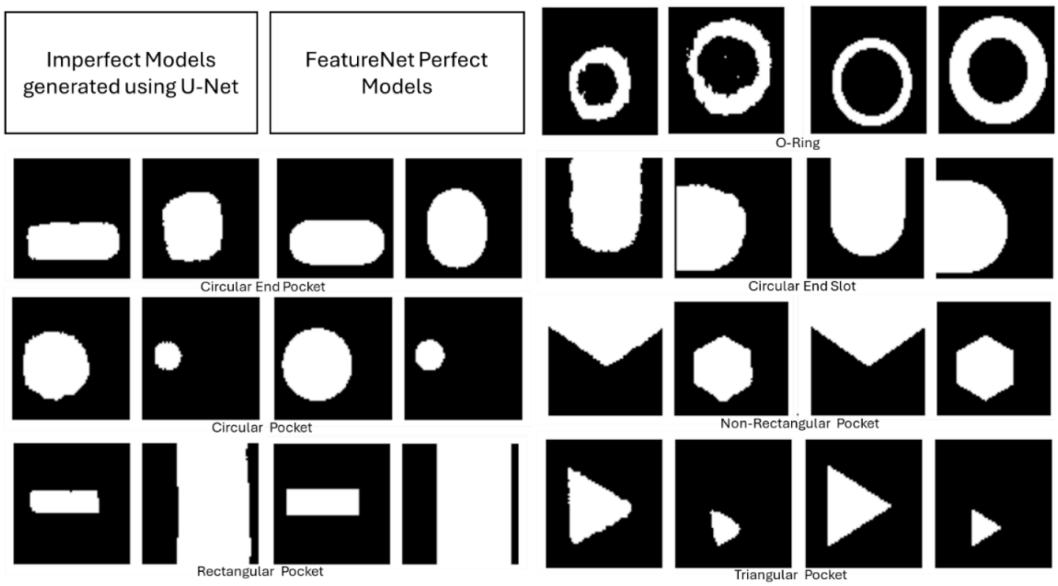


Figure 6. U-Net outputs - imperfect models (left), FeatureNet perfect models (right)

https://doi.org/10.1017/pds.2026.10598 Published online by Cambridge University Press

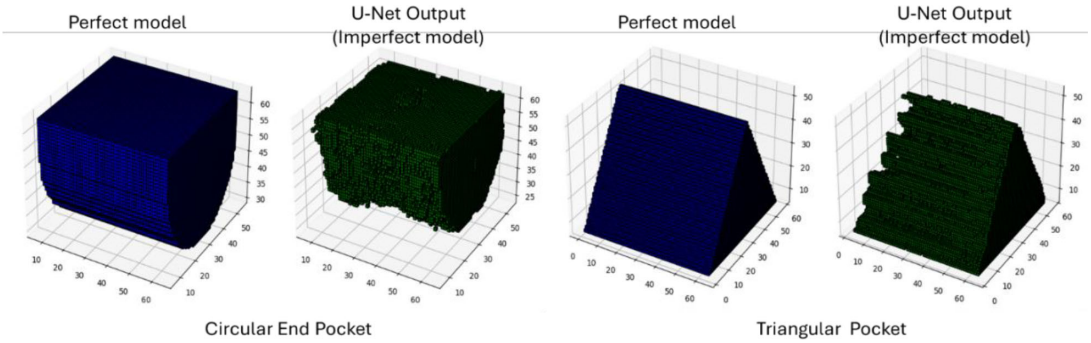


Figure 7. Sample 3D voxel-based models produced by U-Net

4.2. Outcome of trained AFR model

The dataset of perfect and imperfect voxel-based CAD models generated using the above data surrogation module enables us to train an automatic feature recognition model to predict the correct feature type by taking an imperfect model as input.

Ultimately, three trained networks were tested on a dataset of previously unseen imperfect voxel-based models. Five models per category were created manually using VR-Sculpting by the authors, which were then augmented by shrinkage to generate a total of 140 models (20 per category), forming the test dataset. The network trained on D2 (perfect only) performed poorly, giving an accuracy of 65.7%, whereas the one trained on D3 (imperfect only) gave an accuracy of 83.57%. In comparison, a network trained on D1 (both perfect and imperfect) achieved an accuracy of 92.86%. This clearly shows that by including the noise introduced within the imperfect models during training, the systems were better able to correctly recognise features. This holds even in the case where the training set included no perfect models, and had only imperfect data to base its predictions.

Figure 8 shows the confusion matrices of the three networks trained on D1, D2, and D3 when tested on the test dataset. The network trained on D1 struggled with only a non-rectangular pocket (NR). The network trained on D3 was, in addition, confused in recognising Circular end slot (V-CEnd) and NR, confusing it with NR and CEnd. Whereas, a network trained only on a perfect dataset (D2) was confused in recognising Tri, CEnd, while struggling with V-CEnd and NR as well.

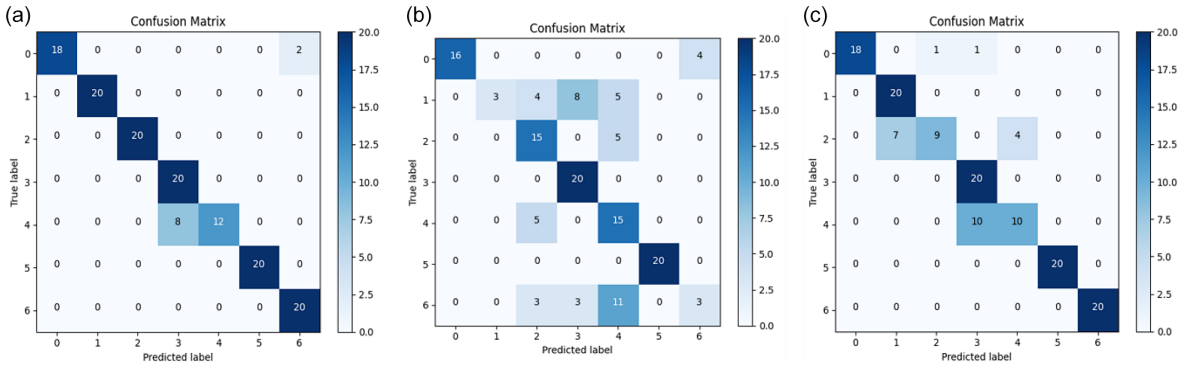


Figure 8. Confusion matrix for (a) D1, (b) D2, and (c) 3

5. Discussion and future scope

The results validate the accuracy of the generated dataset and the novel surrogation method devised in this work. The matching voxel accuracies and dice coefficients of the manually generated imperfect-perfect data pairs with the U-Net generated imperfect-perfect pairs show the effectiveness of the proposed surrogate data generation pipeline in replicating human-like imperfections. Voxel accuracies of all feature types over 0.9 suggest that the trained U-Net model preserved the location and size of the feature, whereas a dice coefficient above 0.6 indicates that the generated output adds a substantial amount of noise in the perfect data.

The testing of the AFR model further shows the importance of our contribution by providing testing with imperfect models. The trained AFR model achieved the validation accuracy of 99.42% and test accuracy of 92.86% on previously unseen data. The comparison of the dataset with imperfect models and without imperfect models further demonstrates the effectiveness of the generated dataset. The 27% accuracy drop

https://doi.org/10.1017/pds.2026.10598 Published online by Cambridge University Press

when trained only on perfect data and the 9% drop when trained only on imperfect data indicate that although imperfect data is important for training a feature recognition model, it must be augmented with perfect data. This could be because perfect models provide a uniform structure of the feature shapes, and once the network is trained on them, it learns the overall shape of the features.

The results confirm the viability of the proposed workflow for data surrogation and pave the way for recognising features from imperfect models, similar in composition to those produced by other digital workflows, such as scanning and CAD reconstruction from images. It could reduce the time-consuming and error-prone tasks of noise filtering and mesh reconstruction. This paper focused on VR sculpting and used it as a source of imperfect CAD models to generate data. The high recognition accuracy of 92% and the fact that it correctly detected most features suggest that the proposed method could help create an intuitive CAD modeller that utilises AI to generate parametric CAD models in real-time.

The current work focused only on 2.5D features, but our results show that even simple symmetric objects become hard to detect when imperfections are present. In the future, the dataset will be augmented for full 3D feature recognition and further developed to recognise intersecting features. The voxel-based dataset presented in this work can also be converted into a signed distance field to recognise features while reducing computational demand, as in (Hilbig et al., 2023). It will be interesting to see if it can give similar or better accuracy.

6. Conclusion

This paper presents a comprehensive dataset of imperfect voxel-based models that can support feature recognition for various downstream applications, such as VR-based CAD modelling, Remanufacturing, and Reverse engineering. It presented a new approach to augmenting perfect data to generate imperfect data using a 3D CNN-based U-Net. The trained U-Net was used to create imperfect voxel-based models from a perfect FeatureNet dataset. The generated dataset yielded voxel accuracies within 4% and Dice coefficients within 15% of those in the original imperfect dataset, demonstrating good replication of human imperfections. The ML model trained for feature recognition on the final dataset performed well on the test dataset, achieving an accuracy of 92.8%, compared to 65.7% when trained only on perfect data and 83.5% when trained only on imperfect data. This work opens up a new paradigm in feature recognition research, which is in line with the latest high-speed but low-fidelity CAD pipelines.

Acknowledgement

The work has been undertaken as part of the Engineering and Physical Sciences Research Council (EPSRC) grants - EP/W024152/1.

References

Cao, W., Robinson, T., Hua, Y., Boussuge, F., Colligan, A. R. & Pan, W. (2020, August 17). Graph Representation of 3D CAD Models for Machining Feature Recognition With Deep Learning. Volume 11A: 46th Design Automation Conference (DAC). https://doi.org/10.1115/DETC2020-22355

Freeman Gebler, O., Goudswaard, M., Hicks, B., Jones, D., Nassehi, A., Snider, C. & Yon, J. (2021). A COMPARISON OF STRUCTURED LIGHT SCANNING AND PHOTOGRAMMETRY FOR THE DIGITISATION OF PHYSICAL PROTOTYPES. *Proceedings of the Design Society*, 1(AUGUST), 11–20. https://doi.org/10.1017/pds.2021.2

Hilbig, A., Vogt, L., Holtzhausen, S. & Paetzold, K. (2023). Enhancing three-dimensional convolutional neural network-based geometric feature recognition for adaptive additive manufacturing: a signed distance field data approach. *Journal of Computational Design and Engineering*, 10(3), 992–1009. https://doi.org/10.1093/jcde/qwad027

Kukreja, A., Cox, C., Gopsill, J. & Snider, C. (2026). From imprecision to precision in virtual reality: a machine learning approach to parametric CAD modelling. *Journal of Engineering Design*, 0(0), 1–28. https://doi.org/10.1080/09544828.2026.2616581

Kukreja, A., Trombini, M., Cox, C. & Snider, C. (2025). A knowledge framework of environment reconstruction methods for mixed reality prototype applications. *Proceedings of the Design Society*, 5, 141–150. https://doi.org/10.1017/pds.2025.10028

Pande, S. S. (2011). *Computer Graphics and Product Modelling for CAD/Cam* (1st ed.). Alpha Science International, Ltd.

Prinz, A., Lee, K. Y., Gupta, A., Park, S. & Göhlich, D. (2025). CAD in Virtual Reality: Integration of Solid Modeling in Standalone VR Application. *Proceedings of the Design Society*, 5, 831–840. https://doi.org/10.1017/pds.2025.10097

https://doi.org/10.1017/pds.2026.10598 Published online by Cambridge University Press

Rameshbabu, V. & Shunmugam, M. S. (2009). Hybrid feature recognition method for setup planning from STEP AP-203. *Robotics and Computer-Integrated Manufacturing*, 25(2), 393–408. https://doi.org/10.1016/j.rcim.2007.09.014

Sunil, V. B. & Pande, S. S. (2009). Automatic recognition of machining features using artificial neural networks. *International Journal of Advanced Manufacturing Technology*, 41(9–10), 932–947. https://doi.org/10.1007/s00170-008-1536-z

Vatandoust, F., Yan, X., Rosen, D. & Melkote, S. N. (2025). Manufacturing Feature Recognition With a Sparse Voxel-Based Convolutional Neural Network. *Journal of Computing and Information Science in Engineering*, 25(3). https://doi.org/10.1115/1.4067334

Wu, S., Khasahmadi, A., Katz, M., Jayaraman, K. P., Pu, Y., Willis, K. & Liu, B. (2023). CAD-LLM: Large Language Model for CAD Generation. *37th Conference on Neural Information Processing Systems (NeurIPS 2023), NeurIPS*, 5–7.

Zhang, H., Zhang, S., Zhang, Y., Liang, J. & Wang, Z. (2022). Machining feature recognition based on a novel multi-task deep learning network. *Robotics and Computer-Integrated Manufacturing*, 77, 102369. https://doi.org/10.1016/j.rcim.2022.102369

Zhang, Z., Jaiswal, P. & Rai, R. (2018). FeatureNet: Machining feature recognition based on 3D Convolution Neural Network. *Computer-Aided Design*, 101, 12–22. https://doi.org/10.1016/j.cad.2018.03.006

https://doi.org/10.1017/pds.2026.10598 Published online by Cambridge University Press